Security for Decentralized Health Information Systems‡

Gerrit Bleumer

Institut für Informatik, Universität Hildesheim, Samelsonplatz 1, D-31141 Hildesheim, Germany

Abstract: The increasing mobility of patients demands more and more large scale integrated health care information systems which provide accurate and secure information across countries' borders. Open distributed systems can support a highly decentralized community such as health care. But the market for open information and operating systems hardly provides secure products. This "missing link" is approached by the prototype SECURE Talk that provides secure transmission and archiving of files on top of an existing operating system. Its services may then be utilized by existing medical applications. This is outlined by a suitable example application.

SECURE Talk is an experimental, high speed tool which does not aim at integrating additional mechanisms into existing operating systems or information systems. However, it demonstrates at a friendly user interface the usability and performance of the mechanisms essential for open systems security: enciphering and electronic signature mechanisms.

0 Introduction

Health care is one of the most fundamental needs of society and, hence, it is deeply woven into modern societies. Many **subjects**, i.e., individuals and organisations take part in health care. Patients look for adequate treatment of their diseases and want their personal data to be protected. Physicians, general practitioners, and nurses provide the health care services. Insurances mostly do the financing. National and international organisations constantly watch the population's state of health and try to improve it. All the subjects together are called the ***health community***[1] in the sequel. Obviously, these subjects have their specific interests, which might be conflicting [Bisk2_89].

Well-being of individuals depends upon *trustworthy* and *economically viable* health care *services*. This implies that, e.g., physicians are personally responsible for the treatment they prescribe[2] and patients may choose the physicians they trust[3]. Hence, one basic characteristic of the health community in contrast to, e.g., companies or the military is that its subjects are not and should not be ruled by one central authority. Otherwise the health community would soon face some kind of health care dictatorship.

Chapter 1 translates health community needs into requirements on health care information systems. Chapter 2 sketches how some requirements can be met by the use of cryptographic mechanisms. Chapter 3 and 4 present a prototype that demonstrates usability and performance of such mechanisms. Chapter 5 summarizes the key items and identifies some open questions.

1 Requirements on Health Care Information Systems

Information is the vital link by which the subjects of the health community co-operate. Increasing mobility and transborder co-operation of these subjects have enforced a trend towards large scale

‡ This work was partly funded by the EC project SEISMED.

1 The term "health community" is meant as a sociological rather than a technical term

2 E.g., physicians swear their hippocratic oath before they enter their profession. This oath stresses their personal responsibility for every treatment they prescribe.

3 This freedom of patients to choose their physicians is not common practice in all EC member states.

integrated medical information systems. Facing the severe restrictions of centralized systems, medical information systems begin to make use of open distributed architectures, which achieve financial (lower cost of off-the-shelf computing and communication products), technical (e.g., interoperability, connectivity) and credibility enhancing (no need for one trusted central authority) advantages.

Let A denote the set of potential *subjects* interacting with an information system S. The term subject denotes any person who has access to the system (by means of some interface). Examples of subjects are conventional end users, system administrators, maintenance staff, as well as attackers who tap lines or inject viruses. An information system S may then be specified by a set I of interfaces i_a for each potential subject $a \in A$ and an *attacker model* $\mathcal{A}$. An interface i_a may be defined by a *functional specification* (which output at interface i_a should be produced by given inputs at any interfaces of I) and a *security specification* (which outputs at interfaces of I should not be produced by given input at interface i_a). The *attacker model* of a system describes for each subject $a \in A$ the worst case behaviour of all subjects $b \neq a$ under which the functional and security specifications of i_a still hold. From the viewpoint of a particular subject a, system S is the more trustworthy the "stronger"[4] its attacker model $\mathcal{A}$ is, i.e., the less a's interface specifications depend upon other subjects' behaviour (at interfaces i_b, $b \neq a$).

A *system S* is called *decentralized* if the attacker model for at least each of its end-users a comprises the behaviour of all other subjects $b \neq a$. "Other subjects" includes in particular those subjects having access to central parts (e.g., communication lines, switching centres, mainframes, etc.) of system S.

In health care, some of the characteristic security requirements at end-user interfaces are *long-term authentication*, *confidentiality*, and sometimes *anonymity* [PfPf_92]. As these security requirements refer to individual subjects, they are reasonable requirements only on decentralized information systems. An integrated information system not committing to the above requirements would not be trusted by, e.g., many physicians and, thus, they would not use it or would refuse to disclose personal data to it. The system were neither trustworthy nor economically viable. Hence, decentralized information systems will play an important role in health care.

The above requirements can be achieved by means of cryptographic mechanisms. Obviously, there are other important requirements, e.g., to define which users are authorized to initiate which transactions upon which data (at which time, etc.), availability of services etc. These latter are not considered here.

1.1 Integrity and Authentication

Perhaps the most important security requirement on storing, processing and transmitting medical data is to ensure its *integrity*, i.e., to prevent undetected, unauthorized modification of the data. In general, this is achieved by appending some information to the data, which serves as a proof of *authentication* to the recipient if and only if it is received uncorrupted. Sometimes it is not only required that the recipient himself is convinced that some data is authentic but he should be able to prove this fact to third parties, too. This kind of authentication is called *non-repudiation*. Once, the author of some data has authenticated this data, his authorship will be provable to any recipient.

Physicians and nurses are personally accountable for the treatment they give to patients. Hence, health care information systems must support to authenticate documents about patients in order to archive them over long periods of time (usually 20 to 30 years). The need for authentication also applies to documents being transmitted. Future applications in telemedicine might highlight this need even more: e.g., medical observation of patients in their homes in order to shorten the periods patients stay in hospitals [PfPf_92]. Various forms of electronic signatures are technical solution to this requirement.

---

4 The term "stronger" is quoted because it does not imply a total order on the set of all possible attacker models.

1.2 Confidentiality

Most medical data is highly personal and sensitive. Accordingly, it is to be protected from unauthorized disclosure during transmission as well as during storage [CEC108_81]. The technical solution to achieve this in open systems is to use encipherment mechanisms.

1.3 Anonymity

Telemedicine might allow patients to access medical information systems and/or expert systems. In the past, information technology has brought more and more services directly to end users without moderation by experts (e.g., desktop publishing, home shopping, etc.) In the future, this might also apply to the medical field. E.g., patients might want to anonymously consult expert systems about mental health care, psychiatric and/or psychological advice, etc.

2 Using Cryptography in Medical Applications

If cryptographic mechanisms are used by medical application software they clearly have to be trustworthy and economically viable. Trustworthiness is achieved by using only published mechanisms that have resisted several years of detailed analysis by independent researchers. Furthermore, the implementation of these mechanisms and their management has to be publicly evaluated according to suitable security criteria. Economic viability can be achieved if software implementations of cryptographic mechanisms are used which are run on standard hardware platforms that are already in place in the health care environment. Within the EC's Advanced Informatics in Medicine (AIM) programme, SEISMED[5] explores the feasibility of this approach by developing SECURE Talk — a demonstrator for cryptographic mechanisms. Chapter 2.1 deals with the integration of cryptographic mechanisms into medical applications in general. chapter 2.2 sketches a practical example of how to enhance the security of a specific application by the use of SECURE Talk.

2.1 Integrating Cryptographic Mechanisms into Medical Applications

To put cryptographic mechanisms into every day practice, one has to integrate them smoothly into existing applications. Smoothly means almost transparent to the intended users. The user interface and the performance should not be affected, unless additional security justifies significant alterations.

Naturally, cryptographic mechanisms would be placed within the operating system and/or the network [ISO 7498-2]. The commercial need for secure open systems increases and the market of commercial operating systems and computer networks is beginning to reflect this by utilizing cryptography. But so far, there are no commercial operating systems or networks available that (i) provide enhanced security by asymmetric cryptography and (ii) use only cryptographic mechanisms that are publicly evaluated. And even if such products were available, most health care environments were unable to afford a completely new operating system and/or network in the short run.

This lack motivated the development of a prototype that demonstrates the performance and usability of cryptographic mechanisms. The prototype allows to process files cryptographically and/or to transfer them. I.e., it can encipher, decipher, sign and verify files, or do combinations of these. The internal encryption and decryption rate achieved is 1.1 MBit/s using a hybrid encryption mechanism. The signing and verifying rate is approximately the same using a DES hash function and RSA (at 512 bit modulus length.) This would allow, e.g., to support 16 ISDN D-

5 SEcure Information Systems in MEDicine (SEISMED) develops a framework of guidelines how to develop and maintain secure health information systems throughout Europe.

channels simultaneously. The prototype is a standalone application collaborating with any existing medical application on a file by file basis. See chapter 3 and 4 for more details.

2.2 A Practical Example

An example for a medical application that can make direct use of file encipherment and authentication is OSIRIS [LRGR_93]. This multimodality image manipulation and analysis software was developed at the University Hospital of Geneva. This software is currently used on different hardware platforms (UNIX with X11 and OSF/Motif windowing as well as Apple Macintosh).

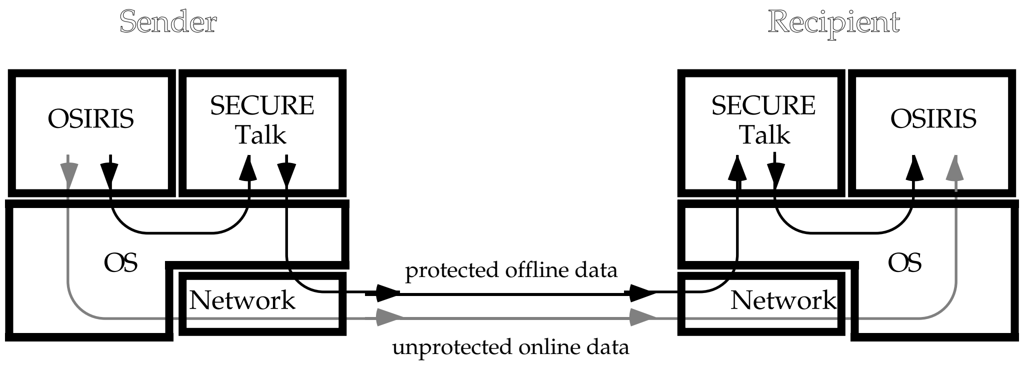

Fig. 1 OSIRIS communication protected by SECURE Talk (Concept)

An extension of this software was recently designed to allow co-operative work on remotely located workstations. During a preparation phase, images are exchanged off-line, e.g., at night. During the consultation phase the next day, a special communication protocol transmits the different actions performed at one station to the other. As OSIRIS utilizes ISDN for communication, the picture files can be transmitted in enciphered form, thus protecting the vast majority of personal data. The consultation data, however, is not transmitted file by file and thus cannot be protected directly by SECURE Talk (Fig. 1).

3 A Prototype for Software Cryptography

The prototype SECURE Talk demonstrates secure communication utilizing only standard hardware. It shows, in particular, which mechanisms yield what performance. The most complex mechanisms are symmetric and asymmetric encryption, message authentication codes and digital signatures.

The message of this prototype is that cryptography (and in particular asymmetric cryptography) is practical for many medical applications even if implemented in software.

The recent years have shown that the computational power of hardware one can buy for one ECU increases by at least 40% each year. This means that software cryptography will fulfil the requirements of more and more ambitious applications at constant cost even if the algorithms are not improved. The prototype presents the operability of software implemented cryptographic services which are integrated into a simplified scenario.

This scenario is interactive, (transparent,) sequential multi-user, secure end-to-end file handling between the workstations[6] of a LAN.

[6] It is explicitly assumed that autonomous workstations are running in the network, not only terminals. These workstations may but need not be equipped with hard disks, CD-drives, smart card readers, etc.

It might, e.g., represent data transfer between the physicians (and/or patients) of different hospitals in possibly different regions or countries. It might, as well, represent the communication within one department of a hospital.

Interactive means that a command interpreter constantly offers a menu of communication and security services to the user, waits for a command, executes it and waits for the next one.

Transparent means that the **security management** is done nearly automatically. This optional feature releases the user from managing parameters and keys as much as possible. In an optimal case, one does not even notice that security services protect the communication.

Sequential multi-user means that multiple users are supported at the same workstation one after another[7]. The underlying operating system is not assumed to support a multi-user mode although this would be helpful. Users are requested to identify themselves before they gain access to the prototype. The basic version of the prototype will support user identification by passwords, future versions might replace this method by more sophisticated techniques like smart cards, biometrics etc. A dependable identification mechanism will be the key to accountability, a necessary property of a system dealing with sensitive data.

Secure end-to-end means that the content of the communication is protected at the application layer [ISO 7498-2] against repudiation of authorship (data authentication), undetected, unauthorized modification (integrity) and unauthorized disclosure (confidentiality). Other network layers are not affected by SECURE Talk.

File handling means that SECURE Talk supports the participants of the underlying LAN in loading, storing, and securely exchanging files created by arbitrary applications. (This includes text–, code–, picture–, video–, biosignal–, sound files, etc.) A generic way to exchange data between applications is by file. Hence, the smallest data unit protectable by SECURE Talk is a file. Many medical applications are supposed to be supportable by this interface, i.e., their communication can be directed into files, which can be transferred by the prototype. Of course, secure file handling in such a generic way does not cover every communication need of *every* medical application. At present, it appears unreasonable to adapt SECURE Talk to more specific data structures.

Beside exchanging other applications' data, SECURE Talk integrates editing and exchanging short text memos. This might be found comfortable not only by users who are used to e-mail.

The hardware assumptions of the prototype reflect the decentralized character of a typical health care environment. It is only assumed that there is some digital, bit transparent network (e.g., ISDN) connecting the workstations available.

Hard- and Software requirements: SECURE Talk 1.0 is implemented for workstations that run Apple operating system 7.0 or higher and are connected by Apple Talk (Ether Talk or Local Talk). On-line features require a set of at least 3 workstations. Off-line features can be demonstrated on at least one single workstation. Only the Apple operating system and the SECURE Talk software have to be installed on each machine. **No extra hardware, especially no crypto hardware is needed**.

Features: All symmetric and asymmetric cryptographic mechanisms are implemented and available as software modules. Characteristic parameters (security parameters, mode of operation, etc.) of each mechanism may be modified by a user as they are subject to the user configuration. This allows high flexibility and adaptability to future needs.

Among the available mechanisms are: Encipherment mechanisms like DES [DES_77], G-DES [PfAß_90], RSA encipherment [RSA_78][8], electronic signature mechanisms like RSA, ElGamal [ElGa_85] and Damgård [Damg_88] signatures, DSS [DSS_92], the crypto-world's first provably

---

7 This is a common requirement for many workstations of a hospital information system, e.g., ward PCs.

8 To avoid active attacks, RSA encipherment is improved by a redundancy predicate.

secure[9] electronic signature mechanism GMR [GoMR_88], etc.[10], hash functions based on any blockcipher available and cryptographically collision free ones [Damg_88] and cryptographically strong pseudo-random number generators like the one presented by Blum, Blum, and Shub [BlBS_86].

4 Concept and Services of SECURE Talk

The prototype consists of one main application –**SECURE Talk**– that is run by each participating workstation of a LAN. SECURE Talk, as a layer 7 application [ISO7498-2], can be structured into 4 sublayers $\boxed{7.0}$..$\boxed{7.3}$[11] of application programming interfaces (APIs) which are described below (also refer to Fig. 2). Control is by an event driven command interpreter.

$\boxed{7.3}$ is the front-end, a graphical user interface (GUI) that presents the SECURE Talk menu to the user.

$\boxed{7.2}$ defines the real SECURE Talk engine that provides secure end-to-end communication. This includes confidential and/or authentic communication. One goal is to achieve accountability among users. The prerequisite for this, however, is that users are supported identifying each other. This implies that, e.g., users reading sensitive data from the system must identify themselves, while sensitive input data should automatically be signed by the sender's equipment. Thus, an additional service of layer 7.2 is the **request for identification**. Version 1.0 provides a password mechanism and requires identification every time layer 7.2 is initialized, i.e., when SECURE Talk is launched.

Correct identification is also a prerequisite for a user specific SECURE Talk configuration. Such a configuration comprises a data folder, a list of addresses and public keys, as well as customized switches and parameters. The configuration can be stored persistently to be used in later sessions (sequential multi-user property of SECURE Talk).

There are two **crypto management modes**: standard and experimental. In **standard** mode, most of a user's configuration is fixed or updated and altered automatically. In **experimental** mode, nearly all of the configuration can be manipulated by the user himself at pleasure.

$\boxed{7.1}$ provides the key management. In particular, distinguished name service, key distribution and certification authority [ISO 9594-1..8]. In contrast to the Kerberos™ network authentication service [BeMe_90, KoNe_92], the key management utilizes asymmetric cryptography in order to provide digital signatures and to avoid all-powerful key distribution centres. There are three **key management modes**: server, client[12], and independent[13]. In **server mode**, the active parts of layer 7.1 are provided: distinguished name service, key distribution and certification authority [ISO 9594-1..8]. However, in server mode the secure communication services of layer 7.2 are not available. In **client mode**, SECURE Talk acts as a client. It assumes there is a dedicated key server available on the network such that all key management is done with that key server. In **independent mode**, SECURE Talk provides the union functionality of server and client

9 Provably secure here means: provable under the assumption that factoring large integers is hard. No other unproven assumption is needed in contrast to many other electronic signature mechanisms like RSA, El Gamal, DSS, etc.

10 Other mechanisms are being implemented.

11 The numbering 7.*i* indicates that these layers are placed on top of OSI layer 7, with 7.*i*+1 using the services of 7.*i* for every *i*=0..2 according to OSI conventions.

12 The terms server and client are short for key server and key client. The only layer where a key client depends on a key server is 7.1, key management. For example, key clients are able to communicate with other key clients at layer 7.0 without involving any key server!

13 The term *independent* stresses the ability of providing secure communication even without any key server.

mode, i.e., the full functionality of layer 7.2 and 7.1. Hence, in independent mode SECURE Talk can act as a client, or as a server, or as both at the same time.

Once, a SECURE Talk application is launched, the user is asked to select a key management mode. That mode cannot be changed afterwards and is hence a characteristic of the running application, i.e., a SECURE Talk process. In the sequel a SECURE Talk process is called Server, Client, or Independent according to the key management mode in which it was started.

7.0 is the back-end and provides the communication primitives (the classical OSI application layer). SECURE Talk applications may communicate with each other in two **data transfer modes**: on-line and off-line. **On-line** means that one SECURE Talk process directly communicates to another SECURE Talk process[14]. **Off-line** means that data transmission is done by another process. A SECURE Talk application that wants to send a message file to some other recipient (e.g., one that is not available on-line) instead passes this file to a process that is capable of communicating with the required recipient. That process might be more flexible and could, for example, allow to transfer files via E-mail, ftp, diskette exchange etc.

Of course, any layer can additionally call standalone services of the underlying operating system[15].

Chapter 4.1-4.4 present the basic services of the APIs 7.2 and 7.1 from different points of view. Chapter 4.1 outlines access control. Chapter 4.2 presents some aspects of key management in more detail, chapter 4.3 presents the crypto management.

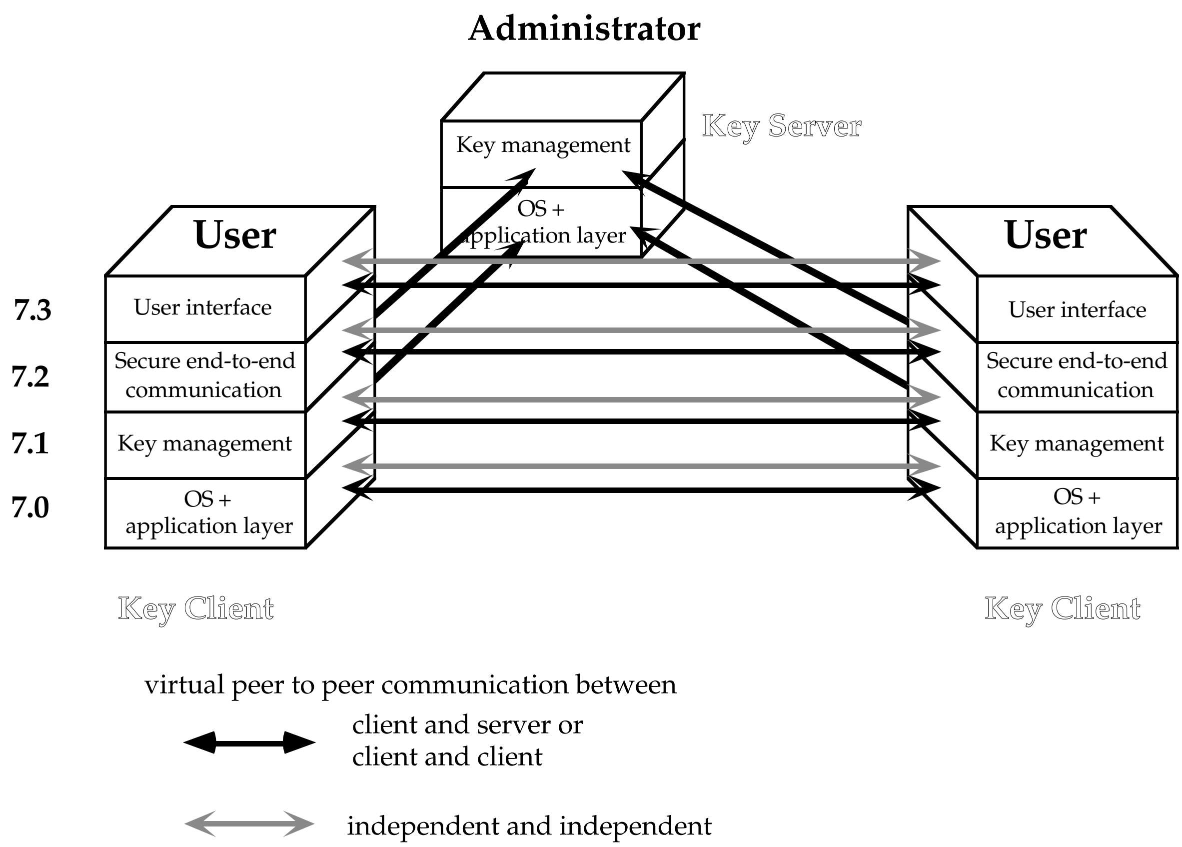

Fig. 2 Structure of the SECURE Talk layers

14 This may be done by using the underlying LAN (e.g. Apple Talk).

15 The Apple System 7.0 and higher integrates basic communication primitives and standalone services like I/O, file-, memory-, and process management.

4.1 Access Control

Users want to have their data managed only by (sub-)systems that conform to the users' personal security specifications[16]. SECURE Talk reflects this requirement by adopting discretionary access control for data owned by individuals. See chapter 4.1.1. The special aspect of audit is considered by chapter 4.1.2.

4.1.1 Discretionary Access Control

Launching SECURE Talk creates a new process under the actual operating system. This process owns and manages a specific domain of data (files and folders). This domain must not be accessible by other, unauthorized users or processes (multi-user operating system). Users of SECURE Talk can define access rights to their own domain of data and grant these to or revoke these from other users (discretionary access control). SECURE Talk does not support to enforce a global security policy and, hence, provides no mandatory access control.

Obviously, SECURE Talk processes running on the same workstation cannot effectively protect their domains against each other if the underlying operating system, such as the Apple operating system, does not (no access control at all, *deleting* files does not imply to *erase* the data, bootstrap from diskette possible, etc.). These weaknesses are considered to be weaknesses of the operating system not of SECURE Talk itself.

Dealing with the specific weaknesses of the Apple operating system, SECURE Talk provides some measures to support users in protecting their sensitive data. For example, each SECURE Talk process emulates some protected persistent memory by requesting a personal diskette from its user. During a session, SECURE Talk keeps track of all secret keys a user generates and all deciphered files, etc. If one attempts to logout one is warned unless all this sensitive data is erased from memory accessible by other processes, i.e., hard disks and RAM.

4.1.2 Audit

A practical system is unlikely to be completely protected from any kind of unauthorized action or misuse of its users. This is because the definition of 'unauthorized' or of 'misuse' is highly context dependent and there is no hope to ever formalize it completely. Of course, there are rules and heuristics about what is definitely or likely to be an authorized or unauthorized action. But there are and will always be conflicts, for example, in which the rights of two patients have to be balanced against each other, which is most appropriately decided by the physician in charge, acting in a responsible way.

The decision of what had been responsible in a certain situation and what had not been, must be left to jurisdiction, finally. This, however, requires investigation after the fact. The canonical way to support such investigations is an audit trail that records every relevant event.

The key questions of audit are i) what are the relevant events, parameters and results [Davi_92], ii) how is the audit trail stored and iii) who holds access rights to read, check, maintain (and delete) log files. In a centralized environment many of these tasks were left to the security administrator of the system. In a decentralized setting the above audit tasks can be decentralized, too. E.g., iii) could be achieved by protecting log files by threshold schemes. In this case, a log file could only be read by a user group of pre-set minimum size. The details need further investigation.

SECURE Talk 1.0 will not support the feature of audit trail for two reasons[17]. (1) Audit trail is much more an organisational rather than a cryptographic measure and hence is only a marginal goal of demonstration for SECURE Talk. (2) A reasonable solution of audit trail in a decentralized environment is still to be designed.

---

16 In HCEs, however, there are also global security specifications that all users have to conform to. This is outside the scope of demonstration of SECURE Talk.

17 However, future versions of SECURE Talk might support log files for audit trail.

4.2 Key Management

The task of key management in general is to provide the right key to the right person. In order to allow decentralized key management, each SECURE Talk configuration maintains its own hierarchy of known recipients and corresponding keys. The user of a process is free to organize this hierarchy according to his needs. Chapter 4.2.1 introduces the general concept that supports symmetric as well as asymmetric key management.

In order to utilize the advantages of asymmetric cryptography, asymmetric key management has to be maintained. Asymmetric cryptography can tolerate that, e.g., persons get to know arbitrary public keys, but it must be guaranteed that public keys belong to the person claimed. The key technique to achieve this is to certify public keys.

The recipient *R* of a public key not only receives the key of a desired participant *S*, but additionally receives a key certificate that he can check for validity. This presupposes that at some time before, *R* has got a public verification key from *S* in a way that he trusted it were indeed *S*'s verification key. For example, all receivers might trust in one central authority and delegate their personal certification authority to it. This kind of key management is called **client—server** in the sequel (chapter 4.2.2). Alternatively, *R* could meet *S* physically, or trust a self-chosen courier. This is called **independent** key management in the following (chapter 4.2.3). These are the two modes of asymmetric key management SECURE Talk offers.

It is possible to run clients and independents within one LAN. In this case, clients publish and receive keys through the server, whereas the server and the independents can exchange keys directly. In a sense the server serves as a key gateway from the domain of independents to the domain of clients.

4.2.1 Concept of Key Storage

Keys are stored in files. One file always contains one key; this may be a secret, a public or a private key. SECURE Talk expects and maintains a predefined structure of key directories to support users in keeping their keys under control: In each user's configuration directory, a sub directory KEYS is expected to exist and to contain at least 3 subdirectories: GENERATED, RECEIVED, and PUBLISHED which may be subdivided by user-defined subdirectories.

GENERATED	contains all secret (asymmetric) and private (symmetric) keys that the user has generated during SECURE Talk sessions.
RECEIVED	contains all public (asymmetric) and private (symmetric) keys that the user has received from other users.
PUBLISHED	contains the public keys corresponding to the secret keys held in GENERATED.

SECURE Talk by default expects and positions each key in one of these sub directories. A 'clean up' tool could be provided that properly restructures a corrupted KEYS directory on demand. All keys contained in the user's home directory are collected, analysed and properly redistributed into the sub directories of KEYS.

4.2.2 Client — Server

One approach to provide recipients with verification keys is to have a central authority, called certification authority (CA), that provides the default trusted path from *S* to *R* [ISO 9594-8]. Trust in the certification authority means that all recipients trust that the CA never produces a signature for a corrupted pair of name and public key. This approach leaves no choice to *R* which path to trust because there is only one.

In this scenario the key distribution service is a (possibly distributed or hierarchical) public directory that provides a list of recipient names, corresponding public keys and the CA's signature for each pair.

To authenticate senders who want to have their public keys certified and registered, a distinguished name service is needed. Its task is to sign a sender's name if and only if it is unique relative to all other senders. To get a key certified, a sender has to provide his name to the CA and, additionally, the corresponding signature received beforehand from some distinguished name centre.

In general, each of the three services (distinguished name, key distribution, key certification) can be provided at a different site within the network.

SECURE Talk offers the functionality of senders and recipients, if run in client mode. In server mode it offers the functionality of distinguished name, key certification, and key distribution by one application. This appears to be a reasonable restriction, if there are very few participants in the network as is the case with the prototype. Future versions of SECURE Talk might moderate this restriction such that decentral naming, public key certification, and key distribution according to [ISO 9594-8] become possible. This would also moderate the requirement that all participants have to trust the same central authority and would additionally yield higher availability because of parallelism and fault tolerance.

As SECURE Talk is initialized in one of the above modes, it is not possible to change the mode during one session[18]. The characteristic of a client is that its name is authenticated and registered by and only by the central server. To initialize a client, it is hence required that a server has been initialized before. It is then necessary to have the client's name registered once and for all at that server. Hence, this is done during the initialisation of a client.

4.2.3 Independent

The idea of independent key management is to allow every participant to act as name, certification and key distribution centre. This enables recipients to rely on those service providers whom they really trust [Zimm_92].

The characteristic of an independent is that its name may be authenticated and registered by other servers and/or independents during a session. Thus, initialisation of SECURE Talk in independent mode does not require any server and the name of an independent SECURE Talk participant is not fixed by initialisation. Instead, an independent S can have himself registered at any other independent R under any name not already registered by R.

4.3 Crypto Management

Global cryptographic parameters: These parameters specify generation of keys and, hence, parametrize all cryptographic mechanisms. Example: Key sizes.
Specific cryptographic parameters: These parameters specify the operation of a particular cryptographic mechanism and may thus be modified after the mechanism is initialized, i.e., a key is generated. Examples: Mode of operation for block ciphers, hash-functions for signature systems, redundancy predicates for block ciphers, symmetric and asymmetric encipherment scheme for hybrid encryption etc.

In experimental mode both kinds of parameters can be arbitrarily specified, whereas in standard mode both kinds of parameters have to be agreed and fixed before SECURE Talk is launched.

5 Summary and Items of Future Interest

The prototype SECURE Talk demonstrates secure communication utilizing only standard hardware. It shows, in particular, which mechanisms yield what performance.

The message of this prototype is that cryptography (and in particular asymmetric cryptography) is practical for many medical applications even if implemented in software.

[18] This, however, is not security relevant. It is only a restriction of flexibility which can be eliminated in future versions of SECURE Talk.

The prototype SECURE Talk basically provides the cryptographic security services of the OSI application layer at a friendly graphical user interface. SECURE Talk supports to communicate securely on-line via network or off-line. One can investigate more or less decentralized forms of public key management and the performance of many different cryptographic mechanisms: 6 ciphers and 5 electronic signature mechanisms including the crypto-world's first provably secure[19] electronic signature mechanism, GMR. All attacks published in the open literature are reflected, e.g., active attacks against RSA encipherment is discouraged by using a redundancy predicate. The cryptographic parameters of all mechanisms can be adapted to specific security needs.

All mechanisms are software implemented in order to be usable with standard hardware. The performance, e.g., of hybrid encryption and decryption (RSA + DES–PCBC) is about 300 KBit/s. That of signing and verifying is approximately the same using RSA with a DES hash function. The internal speed, without disk accesses etc., is about 1.1 MBit/s.[20]

OSIRIS, a medical application running in a distributed environment, shows how SECURE Talk can protect other applications communication on a file-by-file basis.

Integrated health care information systems, of course, have to conform to a specific security policy. Thus they require additional organizational and security properties that could not be included within SECURE Talk 1.0. It remains a future task to design and elaborate them. However, some of these could be demonstrated within the framework of SECURE Talk as soon as they would be available. Two items are mentioned below.

1.) A comprehensive key management could be developed that reflects server based [ISO 9594-8] and server-less [Zimm_92] concepts and possible compromises between both. Also refer to [Boyd_92, BuAN_90, LuSu_89, LuSu_92, TaHu_90, TaHu_92, MTHZ_92]. Such a key management also would have to define a concept for lifetime of keys that is easy to use.

2.) In a health care environment, personal accountability of physicians and nurses must be supported and enforced. Nevertheless, there might be situations where a patient usually requires a signature from just *any* physician (e.g., to prove medical treatment to his health insurance without revealing further information about his physician and disease). Such a signature proves medical authority for the fact under consideration but does not reveal the identity of the physician who actually signed. If it comes to legal proceedings against the physician afterwards, it should be possible to 'open' his or her signature in order to identify the physician who signed.

    Signatures of this type are called *group signatures* in the literature [ChHe_91, DeFr_92, Fran_90, Hwan_91, Pede2_91]. It appears interesting to explore their usefulness in the health care environment.

Acknowledgement

I would like to thank Birgit Pfitzmann, Joachim Biskup and Andreas Pfitzmann for fruitful discussions. Thilo Baldin did an excellent job on designing and implementing parts of the prototype. The implementational work was supported by Gaute Ålmas, Axel Scharff, Angelika Schmidt, Ulrike Schütte, and Volker Pätzold. I am especially grateful to Daniel de Roulet and Yves Ligier for their kind hospitality at the University hospital of Geneva and for introducing OSIRIS to me.

[19] Provably secure here means: provable under the assumption that factoring large integers is hard. No other unproven assumption is needed in contrast to many other electronic signature mechanisms like RSA, El Gamal, DSS, etc.

[20] All performances refer to an Apple Quadra 950 (MC 68040, 33 MHz, RAM: 20 MB, 80ns). Performances of the other cryptographic mechanisms were not yet available.

Bibliography

AnMi_90 Colin I'Anson, Chris Mitchell: Security Defects in CCITT Recommendation X.509 – The Directory Authentication Framework; Computer Communication Review 20/2 (1990) 30-34.

BeMe_90 S. M. Bellovin, M. Merritt: Limitations of the Kerberos Authentication System; Computer Communication Review 20/5 (1990) 119-132.

Bisk2_89 Joachim Biskup: Protection of Privacy and Confidentiality in Medical Information Systems: Problems and Guidelines; DATABASE SECURTIY, III : Status and Propects, IFIP 1990, pp. 13-23.

BlBS_86 L. Blum, M. Blum, M. Shub: A Simple Unpredictable Pseudo-Random Number Generator; SIAM J. Comput. 15/2 (1986) 364-383.

Boyd_92 Colin Boyd: A formal framework for authentication; Proceedings of ESORICS 92, Second European Symposium on Research in Computer Security, LNCS 648, Springer-Verlag, Berlin 1992, pp.273-292.

BuAN_90 Michael Burrows, Martin Abadi, Roger Needham: A Logic of Authentication; ACM Transactions on Computer Systems 8/1 (1990) 18-36.

ChHe_91 David Chaum, Eugène van Heijst: Group signatures; Eurocrypt '91, Brighton, 8-11 April 1991, Abstracts, 130-134.

Chok_92 Santosh Chokhani: Trusted Products Evaluation; Communications of the ACM 35/7 (1992) 64-76.

CEC108_81 Council of Europe (ed.): Convention for the protection of individuals with regard to automatic processing of personal data; European treaty series n° 108; 1981.

Damg_88 Ivan Bjerre Damgård: Collision free hash functions and public key signature schemes; Eurocrypt '87, LNCS 304, Springer-Verlag, Berlin 1988, 203-216.

Damg_92 Ivan Damgård: Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks; Crypto '91, LNCS 576, Springer Verlag, Berlin 1992, 445-456.

Davi_92 Jon David: LAN Security Standards; Computers & Security 11 (1992), pp. 607-619.

DeFr_92 Yvo Desmedt, Yair Frankel: Shared generation of authenticators and signatures; Crypto '91, LNCS 576, Springer Verlag, Berlin 1992, 457-469.

DES_77 Specification for the Data Encryption Standard; Federal Information Processing Standards Publication 46 (FIPS PUB 46), January 15, 1977.

DSS_92 CACM (some editing): The Digital Signature Standard Proposed by NIST; Communications of the ACM 35/7 (1992) 36-40.

ECMA138_89 ECMA European Computer Manufacturers Association: Standard ECMA-138; Security in Open Systems – Data Elements and Service Definitions; December 1989.

ECMATR46_88 ECMA European Computer Manufacturers Association: TR/46; Security in Open Systems – A Security Framework; July 1988.

ElGa_85 Taher ElGamal: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms; IEEE Transactions on Information Theory 31/4 (1985) 469-472.

Fran_90 Yair Frankel: A practical protocol for large group oriented networks; Eurocrypt '89, LNCS 434, Springer-Verlag, Berlin 1990, 56-61.

GoMR_88 Shafi Goldwasser, Silvio Micali, Ronald L. Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks; SIAM J. Comput. 17/2 (1988) 281-308.

Hwan_91 Tzonelih Hwang: Cryptosystem for Group Oriented Cryptography; Eurocrypt '90, LNCS 473, Springer-Verlag, Berlin 1991, 352-360.

ISO7498-2_89 ISO: Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture; INTERNATIONAL STANDARD ISO IS 7498-2; First edition 1989-02-15.

ISO7498_89 Information processing systems – Open Systems Interconnection –Basic Reference Model – Part 2: Security Architecture; International Standard ISO 7498-2 (1989).

ISO9594-1..8 90 ISO/IEC 9594-1: Information technology - Open Systems Interconnection - Specification - The Directory - Part 1..8; ISO International Standard, First edition 15.12.1990.

ITSEC2_91 European Communities - Commission: ITSEC: Information Technology Security Evaluation Criteria; (Provisional Harmonised Criteria, Version 1.2, 28 June 1991) Office for Official Publications of the European Communities, Luxembourg 1991 (ISBN 92-826-3004-8).

ITSEM_92 informal EC advisory group SOG-IS: Information Technology Security Evaluation Criteria (ITSEM); Draft, Version 0.2, 2nd April 1992.

LRGR_93 Y. Ligier, O. Ratib, C. Girard, P. Rubin, M. Rejmer: A Metropolitan Area Network for Teleradiology and Remote Expert Consultation based on ISDN

LuSu_89 Wen-Pai Lu, Malur K. Sundareshan: Secure Communication in Internet Environments: A Hierarchal Key Management Scheme for End-To-End Encryption; IEEE Transactions on Communications 37/10 (1989) 1014-1023.

LuSu_92 W. P. Lu, M. K. Sundareshan: Enhanced Protocols for Hierarchical Encryption Key Management for Secure Communication in Internet Environments; IEEE Transactions on Communications 40/4 (1992) 658-660.

MTHZ_92 Refik Mova, Gene Tsudik, Els Van Herreweghen, Stefano Zatti: KryptoKnight - Authentication and key distribution system; Proceedings of ESORICS 92, Second European Symposium on Research in Computer Security, LNCS 648, Springer-Verlag, Berlin 1992, pp.155-174.

Pede2_91 Torben Pryds Pedersen: A Threshold Cryptosystem without a Trusted Party; Eurocrypt '91, LNCS 547, Springer-Verlag, Berlin 1991, 522-526.

PfAß_90 Andreas Pfitzmann, Ralf Aßmann: Efficient Software Implementations of (Generalized) DES; SECURICOM 90, 8th Worldwide Congress on Computer and Communications Security and Protection, March 13-16, 1990, Paris, 139-158.

PfPf_92 Andreas Pfitzmann, Birgit Pfitzmann: Technical Aspects of Data Protection in Health Care Informatics; Advances in Medical Informatics, J. Noothoven van Goor and J. P. Christensen (Eds.), IOS Press, Amsterdam 1992, 368-386.

RSA 78 R. L. Rivest, A. Shamir, L. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems; Communications of the ACM 21/2 (1978) 120-126, reprinted: 26/1 (1983) 96-99.

TaHu_90 Anas Tarah, Christian Huitema: CHIMÆRA: A network security model; Proceedings of ESORICS 90, European Symposium on Research in Computer Security, Toulouse, France, October 24-26, 1990.

TaHu_92 Anas Tarah, Christian Huitema: Associating metrics to certification paths; Proceedings of ESORICS 92, Second European Symposium on Research in Computer Security, LNCS 648, Springer-Verlag, Berlin 1992, pp.175-189.

Zimm_92 Philip Zimmermann: Phils Pretty Good Privacy 2.0, User Manual;